UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Master as Investigación e Innovación en Inteligencia Computacional y Sistemas Interactivos

MASTER THESIS

Interpretación de Modelos de Clasificación mediante la Proyección sobre la Frontera de Decisión
Interpretation of Classification Models through the Projection on the Decision Boundary

Author: Irene González Velasco
Advisor: Carlos María Alaíz Gudín

September 2020

Irene González Velasco
Interpretación de Modelos de Clasificación mediante la Proyección sobre la Frontera de Decisión

Interpretation of Classification Models through the Projection on the Decision Boundary

Irene González Velasco

PRINTED IN SPAIN

Agradecimientos

En primer lugar, dar las gracias a mi tutor, Carlos María Alaíz Gudín, por todo el tiempo invertido en la dirección y corrección de esta tesis de máster. Sin su guía, este trabajo no habría sido posible. Y también agradecer al ponente, Alberto Suárez González.

Por otra parte me gustaría dar las gracias a todos mis amigos, que han estado a mi lado todos estos años y me han acompañado en el proceso de la realización de esta tesis de máster.

Por último, agradecerle a mis padres y a mi hermano todo lo que han hecho por mí desde siempre, sin su apoyo incondicional, no habría llegado hasta aquí.

Resumen

El aprendizaje automático está presente en prácticamente todos los aspectos de la vida cotidiana, por lo que, debido a su uso generalizado, surgen problemas como su interpretación. Mejorar la interpretabilidad de estos modelos puede ayudar a mejorar la comprensión de los usuarios finales acerca de estos mismos modelos y de los resultados que se obtienen.

En este trabajo de fin de máster, primero se realizará una revisión del estado del arte del aprendizaje automático interpretable a través de varias definiciones y propiedades. Una vez los conceptos básicos hayan sido presentados, se detallarán tres ejemplos de modelos potencialmente interpretables, seguidos de un análisis en detalle de LIME, una herramienta agnóstica para interpretar clasificadores.

Después, se propondrá un método para interpretar la clasificación de un punto concreto a través de su proyección en la frontera de decisión y la diferencia entre ambos puntos. Esta diferencia entre ambos puntos aporta información acerca de por qué se ha clasificado en la clase que se ha clasificado y no en la otra clase. Aunque el método es agnóstico, primero se desarrolla el caso trivial del modelo lineal en el que se obtiene que la diferencia lleva a la interpretación clásica de un modelo lineal. Después, este método se lleva al caso general no lineal aproximando la proyección.

Una vez implementado el método propuesto, se realizarán experimentos con tres conjuntos de datos diferentes y se analizarán sus resultados tanto por separado como con los resultados obtenidos utilizando LIME. Los resultados obtenidos mediante el método propuesto son muy prometedores y coinciden con los resultados esperados al desarrollar el modelo teórico. También se ha comprobado que son similares a los resultados obtenidos empleando LIME.

Palabras clave

Aprendizaje Automático, Aprendizaje Automático Interpretable, Frontera de Decisión, Proyección, Explicación, Interpretabilidad.

Abstract

Machine learning is present in almost every field of daily life, that is the reason why problems such as its interpretability appears. By improving the interpretability of these models, the understanding of the machine learning model's users about that models can increase.

In this master thesis, a review of the state-of-the-art of interpretable machine learning will be done through various definitions and properties. Once the basic concepts have been presented, three examples of potentially interpretable models will be detailed, followed by an in-depth analysis of LIME, an agnostic tool for interpreting classifiers.

After that, it will be proposed a method to interpret the classification of a concrete point through its projection on the decision boundary and the difference between them. This difference between both points provides information about why a point has been classified as a certain class and not as the other class. Although the method is agnostic, first it is developed the trivial case of the linear model, where the difference leads to the classic interpretation of the linear models. Then this method is extended to the general non-linear case by approximating the projection.

Once the method proposed is implemented, three experiments will be done with three different datasets, and their results will be analyzed by themselves and compared against the results obtained using LIME. The results obtained using the proposed method are very promising and match the expected results when describing the theoretical model. They are also very similar to the results obtained using LIME.

Keywords

Machine Learning, Interpretable Machine Learning, Decision Boundary, Projection, Explanation, Interpretability.

Table of Contents

Lists

List of codes

List of equations

List of figures

List of tables

1 Introduction

1.1 Motivation

Machine Learning (ML), the set of models and algorithms which make predictions from some input data, is currently becoming more and more important. Nowadays, this type of models are present in every aspect of daily life, from tools designed to help in the medical field [1] to the recommendation systems [2] implemented in almost every streaming service. With the technological advances that make possible the access to big datasets and great computational power, the use, implementation and access to these algorithms have been democratised, i.e, every day it becomes easier for anyone with programming knowledge to implement, train and use an ML model even in their own homes.

Since the popularity of the ML algorithms has quickly increased and are used in critical areas such as the decision making process in big corporations or even the automation of repetitive tasks in factories, the need to explain and interpret the predictions and their behaviour in an easy and quick way has appeared. The Interpretable Machine Learning (IML) is precisely the branch of ML which tries to interpret those behaviours and predictions in order to increase the trust of the users, gain better comprehension of the models and ensure that they behave as they are intended.

This work is framed within the context of IML, providing a review of the current state of the art and proposing a new approach for the interpretation of binary classification models.

1.2 Objectives

The objective of this work is to propose an IML model-agnostic method that interprets a concrete prediction by analyzing the difference between the instance of the data used to make the prediction and its projection on the decision surface of a binary classifier. This general objective can be split into more specific ones:

- Review the main state of the art in IML in order to obtain detailed knowledge about its current condition and the methods available, studying the different definitions and taxonomies.

- Make an exhaustive analysis of the state-of-art model-agnostic framework Local Interpretable Model-agnostic Explanations (LIME).
- Propose a new model-agnostic method that allows to interpret individual predictions of a binary classifier based on their projection on the decision surface.
- Implement the proposed method.
- Illustrate experimentally the behaviour of this method and compare its results with those of LIME.

1.3 Structure of the document

This master thesis is structured as follows:

Chapter 1 Introduction This chapter motivates the work and summarizes its objectives.

Chapter 2 Interpretable Machine Learning This chapter makes a brief introduction about IML, its importance and the motivation for this work. Then it explores some definitions of IML and a taxonomy. Finally, it shows three examples of potentially interpretable models and analyzes in detail the framework LIME and its relevance for this work.

Chapter 3 Projection on the decision boundary In this chapter a new interpretation method for binary classifiers is proposed. After the motivation, the framework is first developed for linear models, and after that it is extended to any non-linear model.

Chapter 4 Experiments This chapter illustrates the use of the projection method implemented in the previous chapter through some experiments for both linear and non-linear models, comparing the results with LIME. The proposed method will be first applied to a toy two-dimensional example and then its behaviour will be shown in both a low dimensional dataset and in a high dimensional dataset.

Chapter 5 Conclusions and future work This section contains a brief summary of all the work done and the results obtained, and it proposes some related lines of future research.

2

Interpretable Machine Learning

2.1 Introduction

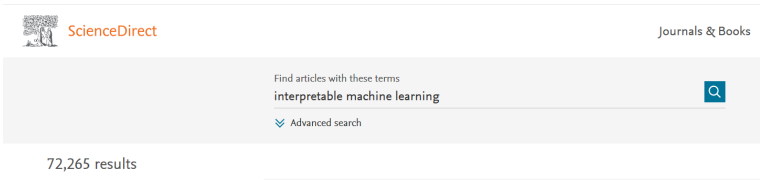
As said in section 1.1, there are Machine Learning (ML) algorithms present in almost every area and they are used daily to perform many different tasks. As they have been used more and more in critical areas, a problem inherent to all ML model has appeared: the difficulty to interpret the model's behaviour and its prediction. For example, algorithms like the ones used in medical diagnosis need to have mechanisms to help the user trust the correctness of the predictions in order to use that information for their decision making process. In other areas of ML like recommending systems it is less important that the user knows that the algorithm makes no mistakes because there are no significant consequences beyond recommending something that the user may not be interested in.

The interpretability and explainability are two characteristics of a model that helps the users trust and truly understand *why* a prediction was made and *why* the model is behaving in a certain way and not in another. This is useful for example in the scientific field, where the discoveries can be disguised as malfunctions of the algorithms but if the unexpected predictions are interpreted, they may lead to interesting advances.

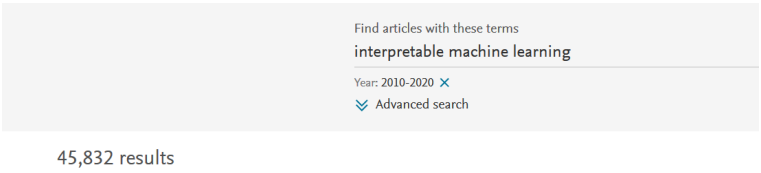
The presence of ML in almost every field and the need of Interpretable Machine Learning (IML) in certain areas have lead to an increasing interest of the subject. In the last 10 years, the number of publications regarding IML has been multiplied by 8 in the database *Science Direct* as can be seen in table 2.1. In figure 2.1 it can be seen that the majority of the publications regarding IML in *Science Direct* has been made in the last decade. These results can also been seen in figure 2.2(c) where the number of publications for the query *interpretable machine learning* has increased exponentially-like in the last years. Figure 2.2 also shows that the majority of publications in the *ACM* database regarding IML has been made in the last decade, which is consistent with the results obtained in the *Science Direct* database.

2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
1,734	2,091	2,354	2,654	3,025	3,675	3,907	4,523	5,742	7,472	8,667

Table 2.1: Number of publications about IML per year between 2010 and 2020 on the Science Direct database.



(a) Query for *interpretable machine learning* without time restrictions.



(b) Query for *interpretable machine learning* restricted to the last decade.

Figure 2.1: Number of publications in the Science Direct database obtained for the query *interpretable machine learning*.

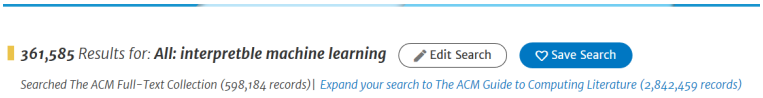
2.2 Definitions and properties

The IML has a fundamental problem: it does not exist a formal definition of *what* is IML, so there are multiple alternative definitions depending on the author [3], although they all present similar ideas:

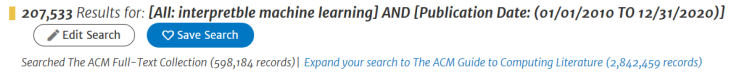
- In the context of ML, interpretability can be seen as the ability to explain or present the ideas in comprehensible terms to the users [4].
- The goal of interpretability is to describe the system with a vocabulary that the user is able to understand [5].
- Interpretability is the capacity of a user to understand the cause of a prediction and to be able to make consistent predictions by themselves [6].

All these definitions have the human user as center of the interpretability since the human understanding is the principal barrier in the process of achieving interpretability. In the context of this work, IML can be understood as a set of models and algorithms which presents their predictions to the users in a comprehensible way.

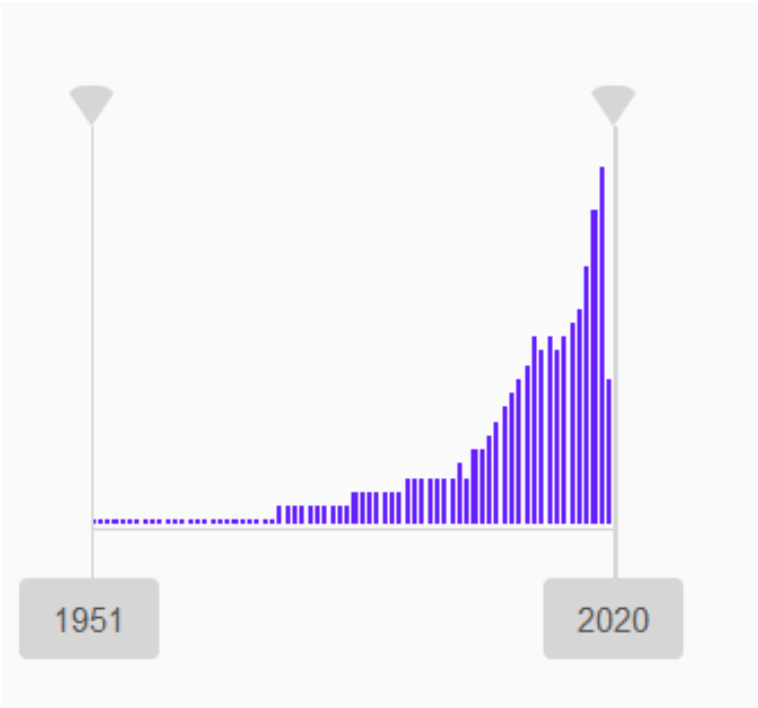
Sometimes, the interpretation alone may not be enough for a particular user, who may also need to obtain explainability. An explainable model is a model with the ability to summarize the reasons of a certain behaviour and the causes of their decisions [5], in other words, that provides explanations. An



(a) Query for *interpretable machine learning* without time restrictions.



(b) Query for *interpretable machine learning restricted* to the last decade.



(c) Number of publication each year since 1951.

Figure 2.2: Number of publications in the ACM database obtained for the query *interpretable machine learning*.

explanation can be seen as a social interaction between a model and a user which creates a common meaning in which the model tries to convince the user that it is capable of completing the task assigned [6]. These explanations are useful when there are incompletitudes in the problem definition [4], which become big barriers in the interpretability. Once again, the important factor of the definition is the user because an explanation should met the expectations of the users and focus on their needs.

Once the concept of interpretation has been introduced, it is interesting to see the main desirable properties of a good explanation method [6, 7], which are:

Expressiveness It is the richness of the structure generated by the explanation method. An explanation has to be interpretable, which means that the explanations should be easy to understand and need to be adapted to the target user.

Faithfully An explanation needs to be, at least, locally faithful. This means that it may be impossible to explain the global model, but at least the explanation must cover the local behaviour in the neighbourhood of the instance being explained.

Portability The explanations should be model agnostic, i.e, an explanation should be able to explain any model.

Translucency It is how much of the explanation method relies on observing into the original model.

There are a set of quality criteria that the individual explanations should follow in order to be a good explanation [6]:

Accuracy How well the explanation is able to explain unknown data instances.

Fidelity How well it explains black-box predictions.

Consistency How much difference exists between models trained for the same task with similar predictions.

Stability How similar are explanations for similar instances.

Comprehensibility How well the user understands the explanations.

Representativeness How many instances the explanation covers.

It exists a trade-off between the interpretation of a model and its completeness [5]. Since the objective of interpretation is to make models easier to understand, the interpretation needs to be simpler enough for the user to understand it, and this can lead to incomplete information about the predictions. In order for an interpretation to be complete and to address all the possible behaviours, the interpretations usually are complex. To avoid this, it is important that the explanations allow a trade-off between completeness and interpretability based on the target user and their needs.

The interpretability of a model can also be used to ensure that the desirable properties of the ML

models are met [4, 6]:

Fairness The predictions of an algorithm should not contain any discriminatory bias, for example to avoid unintentional bias based on race, religion, gender...

Privacy All the sensitive information, such as medical records or personal information, is protected.

Reliability Small changes in the input should not lead to big changes in the predictions.

Causality The algorithms should only pick the causal relationships.

Trust It is easier for the user to trust a system that explains its decisions than to trust a black-box system.

Usability Interpretable algorithms should give enough information to the user to help in their tasks.

As seen before there are some factors that affects the interpretability of a model, related with the circumstances of the final user. They are:

Previous knowledge The users are influenced by their previous knowledge, expectations or beliefs about the model that influence how they interpret the model and if they accept or not an explanation.

Patience of the users The interpretability of a model depends on how much time and effort an user is willing to spend in order to understand the results.

Representation The representation of the interpretable method needs to be adapted to who is going to interpret it since a non-expert user may need a simpler representation than an expert one.

To be able to define correctly the interpretable methods it is important to have a classification that helps to understand their characteristics and their qualities. There are several possible different taxonomies of the interpretation methods based on the criteria chosen to classify them [6]:

- Based on how they achieve the interpretability:
  - Intrinsic: The interpretability is obtained by restricting the complexity of the model.
  - Post-hoc: After the training it is possible to apply methods that analyze the model.
- Based on the level of the interpretation:
  - Individual: The interpretation is made for individual predictions.
  - Local: The interpretation is made for a specific subset of the input space of the model.
  - Global: The interpretation is able to explain the complete model.

- Based on the specificity of the method:
  - Model dependant: The method is designed to interpret a specific model.
  - Model agnostic: The method can interpret different models.

2.3 Interpretable models

The easiest way of achieving interpretability without any external methods is by using a subset of potentially interpretable models. They are called potentially interpretable because thanks to their characteristics it is feasible for the user to interpret them without external methods. In this section, three of the most common interpretable models are explained: the linear regression models, the models based on decision rules and the models based on decision trees.

2.3.1 Linear regression models

The linear regression models are ML models that predict the output as a weighted sum of the input parameters plus a bias, so they can be expressed as $y = \mathbf{w}^T\mathbf{x} + b$ where $\mathbf{x}$ are the input parameters, $\mathbf{w}$ the weight vector, b is the bias, and y is the prediction [8, 9]. The great advantage of the linear regression models is its linearity because that linearity makes the interpretation easy to understand, so they are methods common in every field where *how* the result is obtained is as important as *what* is the result.

The linear models are well studied and accepted since they are easy to implement and fast calculating their predictions. And since the prediction is a weighted sum, the model is transparent and easy to interpret. For example, if the feature x_i is a numeric characteristic that changes in z units, the output when the value of that characteristic is modified while the rest of values are fixed, changes in $w_i z$ units [6].

The main disadvantage of the Linear Models (LM) as interpretable models is that they do not have all the characteristics of a good explanation that were presented in section 2.2. For example, the interpretability decreases quickly as the number of features of the model increases. This can be fixed with sparse methods such as Least Absolute Shrinkage and Selection Operator (LASSO) [9], that selects the most important features and makes predictions based only on that subset of the inputs. With this type of methods, the number of relevant parameters decreases and so the interpretability increases.

2.3.2 Models based on decision rules

Another potentially interpretable models are the ones based on decision rules. A decision rule is a structure *IF-THEN* which consists in an antecedent, *if* a certain condition or set of conditions are met, and a prediction, *then* a prediction is made [6].

The utility of a rule is determined by the number of different cases that the antecedent cover, which affects directly the interpretability. This means that as the number of conditions in the antecedent increases, the readability and the interpretability of the rule decreases.

In order to obtain a good classification it may be needed to learn a huge number of rules which leads to issues of rules overlapping or cases where no rule is applied. These scenarios decrease the interpretability of the model since the decisions may not be trivial anymore.

The main advantage of the decision rules is that they are easy to interpret, the prediction is quick and usually generates sparse models, what reduces the number of relevant parameters. All these characteristics makes the decision rules a great base from where to build more complex models. One of their main disadvantages is that the parameters are often required to be categorical, but it is also the base of the interpretability power since the categorical parameters are easier to interpret.

The interpretation of decision rules is trivial. Having a list of decision rules in the form:

IF $condition_1$ THEN $prediction_1$

IF $condition_2$ THEN $prediction_2$

IF $condition_3$ AND $condition_4$ THEN $prediction_3$

DEFAULT $prediction_4$

The interpretation is easy. If the $condition_1$ is met, then the result is the $prediction_1$, if the $condition_3$ and $condition_4$ are met, then the result is the $prediction_3$. If none of the conditions are met, then the result is the $prediction_4$ [6]. As the number of rules or the number of conditions per rule increases, the interpretation of the decision rule decreases since it is more difficult to read.

2.3.3 Decision tree models

The decision tree models are similar to the decision rules. These models are based on a tree structure where the data is divided multiple times and, starting from a root node, each division of the tree leads to a prediction [8, 9].

Each individual prediction made in a decision tree can be explained by decomposing the path starting in the root node, each vertex indicating the subset of the data being observed, all the way to the leaf node that predicts the output. This means that the interpretability of the model depends directly

on the length of the tree because as the number of intermediate nodes between the root and the leaf increases, the interpretation of the tree becomes more complex.

The decision trees are able to represent the relationships between its characteristics. Since the data is divided into smaller subsets, the interpretability increases with each division. The tree structure is also easy to visualise for the users so the interpretation is trivial.

They great disadvantage is that they are unstable, so small changes on the training data create completely different trees. Moreover, their interpretability is dependant on the depth of the tree and quickly decreases.

In figure 2.3 is shown an example of a decision tree. The interpretation, starting from the root node, is similar to the decision rules. For example, assuming that *Condition1* and *Condition3* are met, the result of the tree would be *Prediction1*.

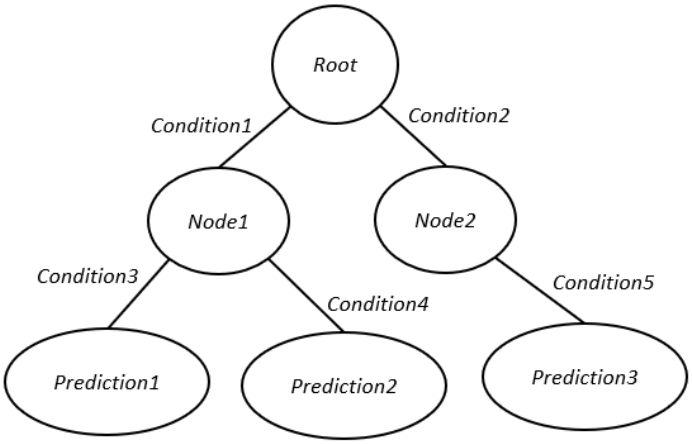


Figure 2.3: Example of a decision tree.

2.4 Local Interpretable Model-agnostic Explanations: LIME

As seen in section 2.2, the agnostic methods have the advantage that they can be used to interpret multiple different models. In this section the tool Local Interpretable Model-agnostic Explanations (LIME) [7] is analyzed in detail since it is currently one of the state-of-the-art agnostic methods, as it can explain a prediction of any classifier by learning interpretable local models around that prediction.

LIME was proposed because, despite how extended the use of ML algorithms are, the majority of users still see them as black-box models. This can make the users not to trust neither the prediction of the model or that the model would behave as intended and, as a consequence, to avoid using that model. As explained above, this trust may also be influenced by the previous knowledge that the users may have and that makes them to accept or reject a prediction depending on whether they understand

the reasoning behind that prediction or not.

Besides the trust of the users, another reason to explain a concrete prediction is the need to evaluate the model before deploying it in a real environment. Nowadays, the evaluation of ML models is usually made with a validation dataset and metrics such as the accuracy, but these metrics may not be correct when using the model in the real world since, for example, the accuracy of the models is often overestimated [10]. One solution is to inspect and explain individual predictions in order to complement those metrics to provide a better understanding of the model.

The local explanations also help to find problems with the data such as characteristics strongly correlated that may lead to bad performance of the model when deployed in the real world, that may not be visible by looking to the raw data but may be obvious when being explained.

Another reason to explain locally different predictions in order to comprehend the model is to be able to compare from a set of models and choose the one that performs better. The explanations may not be relevant at first sight since there are metrics that compare classifiers, but, as explained before, the metrics alone are not as useful as they are when accompanied by some explanations.

As seen in section 2.2, there is not a formal definition of interpretability or explainability, so every author proposes one. Following a definition similar to the one of this work, the authors of LIME intend to present textual or visual artifacts that provide understanding of the relationships between the instance and the prediction. These explanations should meet the desirable characteristics also reviewed in section 2.2.

LIME is a tool that provides an auxiliary interpretable representation of an interpretable model that is locally faithful to the original classifier.

An explanation can be formally defined as a model $g \in G$, where G is a family of potentially interpretable models such as the ones seen in section 2.3. Since not every model may be interpretable, it is defined $\Omega(g)$ as the complexity, interpreted as the opposite of interpretability, of the explanation g. The model explained can be denoted as $f : \mathbb{R}^d \to \mathbb{R}$. The distance between an instance z and x is defined as $\pi_x(z)$. The measure of how unfaithful the model g is in approximating the original model f in the locality $\pi_x(z)$ is defined as $\mathcal{L}(f, g, \pi_x)$. Now, the explanation must minimize $\mathcal{L}(f, g, \pi_x)$ while $\Omega(g)$ is still understandable to the user in order to ensure both interpretability and local fidelity. The explanation that LIME produces can be expressed as:

$$\xi(x) = \arg\min_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g). \tag{2.1}$$

Since no assumptions were made, the previous formal definition can be used with different explanation families, fidelity functions and complexity measures.

To learn the local behaviour without making assumptions about the model it is important to make

sure that if the interpretable inputs vary, the method still can approximate $\mathcal{L}(f, g, \pi_x)$. To do this, LIME draw instances weighted by π_x, which means that LIME obtains a random subset of samples in the locality defined by π_x and obtains the labels by evaluating those samples with the f function that is being explained. With this $\mathcal{Z}$ dataset, the previous formal definition of the explanation of equation 2.1, $\xi(x)$, is optimized. This locality can be seen in figure 2.4 where the the function f is represented by the pink/blue background and cannot be approximated by a linear model, the big red cross is the instance being explained and the black dotted line is the learned explanation, which is locally faithful but not globally. The weights of the samples randomly drawn are represented bigger or smaller depending on the distance to the instance that is being explained.

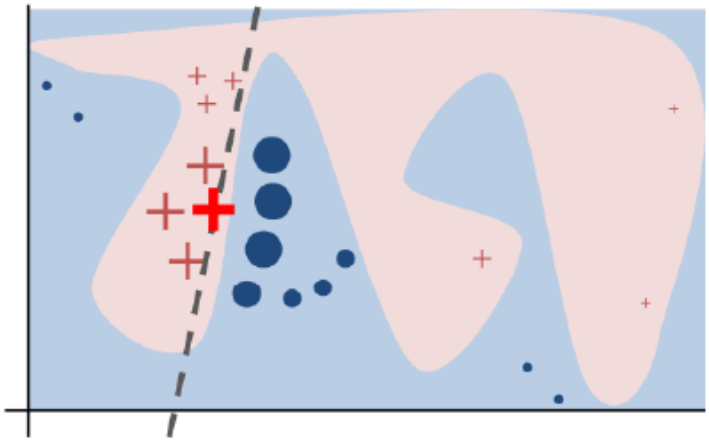


Figure 2.4: Illustration of the locality intuition shown in the original paper of LIME [7]. The instance being explained is the red bold cross, the pink and blue background corresponds to the non-linear function f, the dashed black line corresponds to the learned explanation and the locality is represented by the size of the points which are bigger near the red cross and smaller when far from the red cross.

Another feature that is presented in LIME is a tool to pick the important instances to be explained, but since this tool is not relevant for this work, it will not be explained.

In conclusion, this method provides a tool to explain any model by learning interpretable local models around that prediction. As explained before, this is known as an agnostic-model method and it is desirable because it allows to compare different models in the same conditions.

The method of the projection, detailed in chapter 3, proposes an approach comparable to LIME, able to explain any binary classifier by interpreting single predictions locally. Chapter 4 shows some usage examples of LIME while being compared to the projection method proposed in this work.

3

Projection on the decision boundary

3.1 Introduction

Inspired on the LIME tool explained in section 2.4, the idea is to propose a model-agnostic method that explains a single prediction in any binary classifier.

To know *why* an instance belongs to a certain class it may be useful to know *which* characteristics or, more precisely, *what* values for that characteristics are the ones that make that instance of data belong to one class and not the other. The hypothesis of this work is that the decision made by a model to classify a certain instance can be understood by analyzing the difference between the instance and the nearest sample that the model would not know how to classify. The issue is how to obtain that nearest point. In particular, the decision surface between the two classes can be seen as the set of points that do not belong to any of the classes, since they are the ones that separate them.

This means that, knowing the instance that is being explained and the set of points that belongs to the decision surface, it is possible to obtain the point that the classifier does not know how to classify closest to the instance, by calculating the projection on the decision surface and, with the projection, obtaining those differences.

The only assumptions made on the models being explained is that they follow the interface of the library `scikit-learn` [11], which is a python library that provides useful tools to handle ML models. In particular, the models are assumed to implement the method `predict_proba`.

3.2 Linear models

In the linear classifiers the decision surface and the projections have trivial exact solutions. Assuming the basic case of a two dimensional data set with two linear separable classes, the general idea can be see in figure 3.1.

Specifically, a linear classifier is a basic classification model that can be expressed as:

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b, \tag{3.1}$$

where $\mathbf{x}$ is the input parameter vector, $\mathbf{w}$ is the weight vector and b is the bias of the model. There are three possible solutions for the output y:

- $y(\mathbf{x}) < 0$: The point is classified as class $C1$.
- $y(\mathbf{x}) > 0$: The point is classified as class $C2$.
- $y(\mathbf{x}) = 0$: The point is on the decision boundary and the classifier does not know how to classify that point. Usually, it will be assigned to one of the two classes by default.

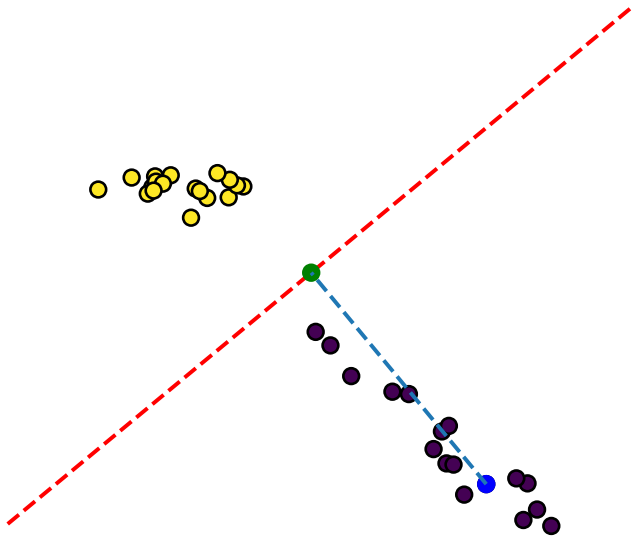


Figure 3.1: Graphic example of the linear two dimensional model. Each color represents a different class, the dashed read line corresponds to the decision boundary, the bright blue point is the instance being explained, the green point is the projection of the instance and the blue dashed line is the distance.

The first step for computing the projection on these models is to obtain the decision surface. The points in the decision surface are the points for which $y(\mathbf{x}) = 0$, so they are the $\mathbf{x}$ points that satisfy:

$$\mathbf{w}^T\mathbf{x} + b = 0. \tag{3.2}$$

The next step is to obtain the projection, $\mathbf{x}'$, of the point $\mathbf{x}$. This projection is the point on the decision surface where the distance between $\mathbf{x}$ and $\mathbf{x}'$ is the minimal distance. This signed distance r can be calculated as

$$r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}, \tag{3.3}$$

where $\mathbf{x}$ is the point, $y(\mathbf{x})$ is the evaluation of equation 3.1 for the point, and $\|\mathbf{w}\|$ is the norm of the weight vector.

The projection $\mathbf{x}'$ is, by definition, the closest point on the decision boundary to the point $\mathbf{x}$ so if the minimum signed distance r is known, the projection can be calculated as:

$$\mathbf{x}' = \mathbf{x} - r\frac{\mathbf{w}}{\|\mathbf{w}\|}, \tag{3.4}$$

where $\mathbf{x}$ is the point that is being projected, $\mathbf{x}'$ is the projection, r is the signed distance between the point and the decision boundary and $\frac{\mathbf{w}}{\|\mathbf{w}\|}$ is the unitary normal vector of the hyperplane.

The difference between the original point $\mathbf{x}$ and the projection $\mathbf{x}'$ is:

$$\mathbf{d} = \mathbf{x} - \mathbf{x}'. \tag{3.5}$$

This difference is proportional to the $\mathbf{w}$ vector which is trivial using the results equation 3.4 and equation 3.5,

$$\begin{aligned} \mathbf{d} &= \mathbf{x} - \mathbf{x}' \\ &= r\frac{\mathbf{w}}{\|\mathbf{w}\|}, \end{aligned} \tag{3.6}$$

so it can be seen that the difference is proportional to $\mathbf{w}$, as it was to be expected geometrically.

Code 3.1: Algorithm for obtaining the exact projection of a point p in a linear model.

```
def get_exact_projection(model, p):
    def get_projection(point, y_value, w):
        return point -(y_value/np.linalg.norm(w))*(w/np.linalg.norm(w))
    def calculate_y(point, w, b):
        return w@point + b

    w = model.coef_[0]
    b = model.intercept_
    y = calculate_y(p, w, b)
    projection = get_projection(p, y, w)

    return projection
```

In listing 3.1 it is shown the algorithm for obtaining the projection in linear models. Once the projection and the differences are calculated, the next step is the interpretation of the results. This interpretation step will be detailed in chapter 4 with an example and a comparison with LIME.

3.3 Non-linear models

Once the results for the trivial case, the linear model, described in section 3.2 are tested and have promising results, detailed in chapter 4, the method can be extended to non-linear models.

The general idea can be seen in figure 3.2. Knowing the difference between a point $\mathbf{x}$ and the closest point that the classifier does not know how to classify, which by definition is the projection $\mathbf{x}'$ on the decision surface, it may be possible to interpret *why* a point has been assigned to a certain class and not to the other. The issue is that the non-linear case does not have an exact solution for obtaining the decision boundary and the projection on it, so it has to be done by approximating both.

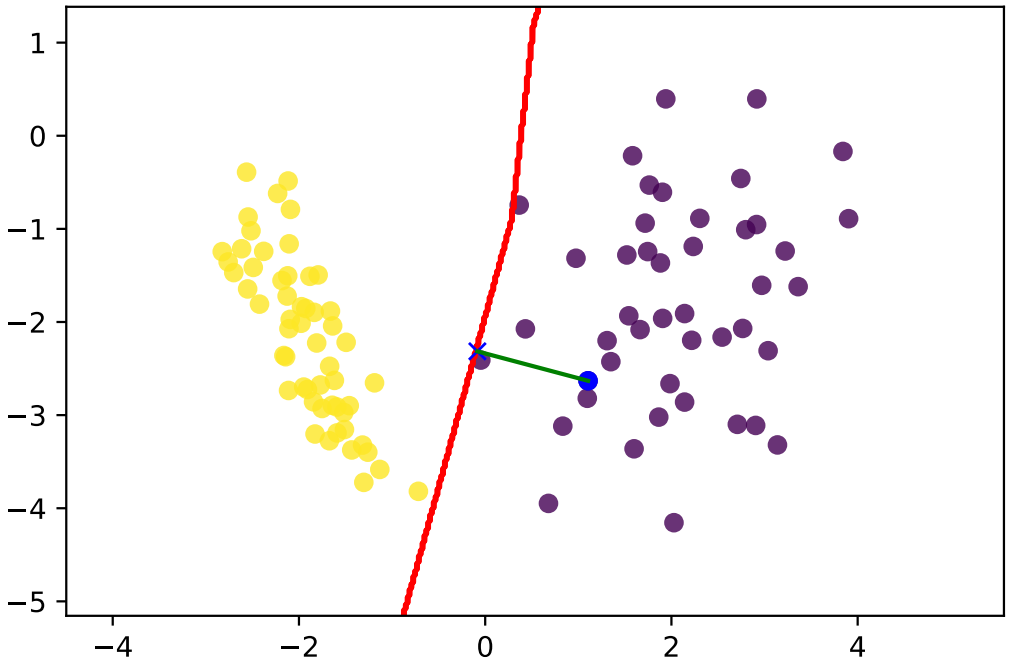


Figure 3.2: Graphic example of the non-linear two dimensional model. The yellow and dark blue points correspond to each one of the classes. The bright blue circle corresponds to the instance being explained, the blue $\times$ is the projection of that instance on the decision boundary. The decision boundary is the red line. The green line is the difference between the projection and the original instance.

The decision boundary can be seen as the points on the space where the probability of belonging to each of the classes is 50%, so the decision boundary may be approximated by looking into the space for the points satisfying this condition.

The process of projecting on the decision surface is made using the library `scipy` [12], in particular the method `fmin_cobyla` [13] which is a method that, building lineal polynomial approximations to an

objective and a set of constraint functions, finds the solution.

In this case, the objective function that needs to be minimized is the distance from the original point $\mathbf{x}$ to the points on the decision function. For convenience, the squared distance would be used instead and is calculated as:

$$\mathbf{d} = \|\mathbf{x} - \mathbf{x}'\|^2. \tag{3.7}$$

The points that are being looking for are the ones that belong to the decision boundary, so the algorithm needs two constraints, p_0 and p_1, one per class.

$$\begin{aligned} p_0(\mathbf{x}') &\leq 0.5, \\ p_1(\mathbf{x}') &\leq 0.5, \end{aligned} \tag{3.8}$$

so the optimization problem is:

$$\min_{\mathbf{x}'} \frac{1}{2}\|\mathbf{x} - \mathbf{x}'\|^2 \text{ subject to } p_0(\mathbf{x}') \leq 0.5 \text{ and } p_1(\mathbf{x}') \leq 0.5. \tag{3.9}$$

Since `fmin_cobyla` guarantees that the value of each constraint is larger or equal to zero, the constraint should be rewritten as $c_0 = p_0(\mathbf{x}') - 0.5$ and $c_1 = p_1(\mathbf{x}') - 0.5$.

It is important to take into consideration that, in some classifiers, when the algorithm reaches remote regions, the data tend to have probabilities of 50% for each class. In order to prevent the optimization algorithm to search for these distant regions, the initial point is initialized using the mean values of the data. The whole implementation can be seen in listing 3.2.

With the projection calculated, the next step is to obtain the difference between the two points:

$$\mathbf{d} = \mathbf{x} - \mathbf{x}'. \tag{3.10}$$

Once the projection and the differences are calculated, the following step is to interpret those results. This interpretation will be detailed in chapter 4 with an example and a comparison with LIME.

Code 3.2: Algorithm for obtaining the approximated projection of a point p in a non-linear model.

```
1  def get_approximated_projection(X, model, P):
2      def c1(x):
3          probs = model.predict_proba([x])
4          return probs[0][0]-0.5
5      def c2(x):
6          probs = model.predict_proba([x])
7          return probs[0][1]-0.5
8      def objective(x):
9          return 0.5*np.linalg.norm(x.ravel()-p.ravel())**2
10
11     x0 = np.mean(X, axis=0)
12     p = P
13     projection = fmin_cobyla(objective, x0, [c1, c2])
14
15     return projection
```

4

Experiments

Once the methods for interpreting linear and non-linear models have been implemented, the next step is to design experiments in order to verify that they work as intended and it is possible to interpret the results of a binary classifier.

The method has been tested in three different scenarios:

1.– A trivial case, where the dataset is generated randomly and has two dimensions.

2.– A case where the dataset has low dimensionality. In this example it is used the dataset *Iris Setosa* [14], which is well known and studied and, although it has three different classes, this experiment only uses the two classes that are linearly separable.

3.– A case where the dataset has high dimensionality. In this example it is used a dataset for optical recognition of handwritten data [15], which is well known and studied. This dataset has ten different classes but the example will only use two of them.

These three different scenarios have two possible cases each one:

1.– Linear model: The exact projection method is being tested obtaining the exact projection and decision boundary of an Logistic Regression (LR) using the class `LogisticRegression` [1] from `scikit-learn`. A LR [8] is a linear classifier which predicts the probability of the input of belonging to a certain class and its interpretation is similar to the interpretation described in section 2.3 since it is also a potentially interpretable method.

2.– Non-linear model: The experiments for the approximated method are going to be done training a Multilayer Perceptron (MLP) [8] using the class `MLPClassifier` [2] from `scikit-learn`. An MLP is a supervised learning algorithm that can learn non-linear functions either for classification or regression since between the input and the output there can be one or more non-linear layers.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[2] https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

The same models will be used in all the experiments in order to facilitate the interpretation of the points.

Notice that the experiments show here are not deterministic, since the selection of the patterns is random, so additional executions of the code could lead to different results, although all of them should suggest identical conclusions.

4.1 Two dimensional data

The first experiment is a trivial case, where the data have two dimensions and can be plotted in a two dimensional space to have a better understanding of the behaviour of the model.

Both experiments follow the same steps:

1.– Create the random datasets.

2.– Separate the data into the train and test sets.

3.– Train the model.

4.– Obtain the projection on the decision boundary.

5.– Explain two points of the model.

6.– Compare the results obtained using the proposed projection method with the results obtained with LIME.

4.1.1 Linear model

For the linear model experiment the data is generated randomly by using the method `make_classification` [3] provided by `scikit-learn`. In this example it was generated a set of 1000 points that are split into train and test sets with the method `train_test_split`, also provided by `scikit-learn`, in a proportion of 90% of the points belonging to the train set and 10% of the points belonging to the test set. The model would be trained with the training set and then the points explained would be points belonging to the test set. The data generated is shown in figure 4.1 where there can be seen two different classes. Once the train and test sets are split, the train set is used to train a LR model.

The interpretation of this method is trivial since there are only two features generated randomly which represents the two coordinates of a two-dimensional space.

Once the model is trained, the next step is to obtain the projection on the decision boundary. In this experiment, this projection is obtained for two different points and the projection can be seen in

[3] `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html`

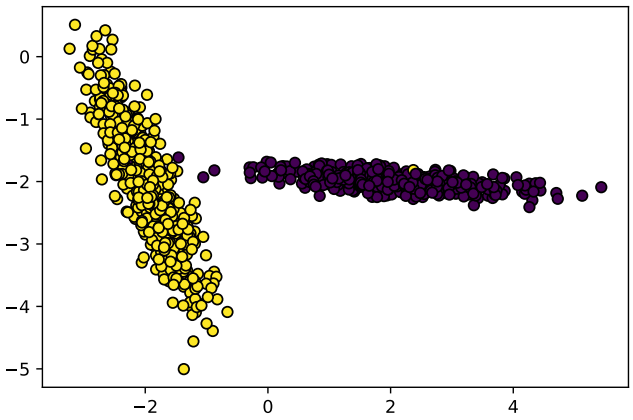


Figure 4.1: Random dataset generated for the two-dimensional linear experiment. There are two separated classes represented by the yellow color for the class 1 and the dark blue color for the class 0.

figure 4.2. The original points are represented with the blue $\times$, the projections are the red $\times$ and the $\mathbf{d}$ vector is represented by the dotted lines. The colored degraded background represents the probabilities of the classes and goes from 0.0 in pink color to 1.0 in green color. In the middle of the image it can be seen a white band that represent the region where the probability is between 0.45 and 0.60, i.e, with the decision boundary. The points projected are exactly on the middle of this band, so the projection on the decision boundary of the two selected points has been done correctly.

Knowing that the projection is correct, the next step of the interpretation process is to obtain the coordinates of the points being explained, the coordinates of the projection and the $\mathbf{d}$ vector.

```
The original point belongs to class 0.
The original point has coordinates (2.03, -2.00).
The projection has coordinates (0.02, -2.92).
The difference is (2.01, 0.92).
```

The results obtained can be seen above. The coordinates for the point being explained are $(2.03, -2.00)$, the coordinates of the projection are $(0.02, -2.92)$ and the difference vector $\mathbf{d}$ is $(2.01, 0.92)$. This results indicate that, in order for a point to be classified as class 0, both coordinates needs to increase its value. In the $\mathbf{d}$ vector, in absolute values, the first coordinate is bigger that the second one, so the most relevant changes are made on that coordinate.

The LIME interpretation of the first point is:

```
[('x > 1.93', -0.680515475038883),
('-2.21 < y <= -2.00', -0.019749694701959312)]
```

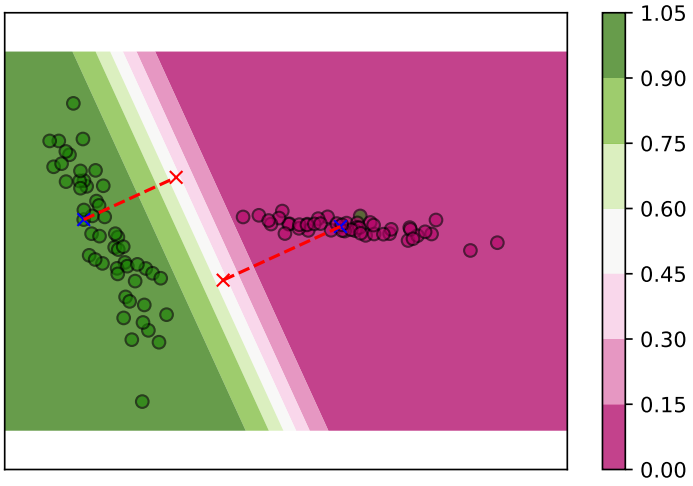


Figure 4.2: Projection of two points in the linear two-dimensional model. The background colors represent the probability of a point belonging to the 1 class, from 100%(green) to 0%(pink). Since there are only two classes, the points with 0% probability of belonging to the 1 class, belongs to the 0 class. The blue $\times$ represents the point that is being explained, the red $\times$ represent the projection on the decision boundary, which is the band white colored that has 50% probability of belonging to the 1 class. The dotted red line represents the difference between the original point and the projection.

The first coordinate, named x, is the feature with bigger predictive power in absolute terms and the second coordinated, named y, has the smaller predictive potential. These results are consistent with the ones obtained in the projection method because the feature with more predictive potential is the one that needs to make bigger changes in its value and has the bigger weight in the $\mathbf{d}$ vector.

The results of the projection method for the second point that is being explained are:

```
The original point belongs to class 1.
The original point has coordinates (-2.38, -1.88).
The projection has coordinates (-0.79, -1.15).
The difference is (-1.59, -0.73).
```

The coordinates of the original point are $(-2.38, -1.88)$, the coordinates of the projection are $(-0.79, -1.15)$ and the difference vector, $\mathbf{d}$ is $(-1.59, -0.73)$. This indicates that both coordinates needs to decrease its value. The absolute values of the $\mathbf{d}$ vector also indicates that the first coordinate is the most relevant when making predictions, since it has the biggest value.

When explaining the second point with LIME, the results obtained are:

```
[('x <= -1.95', 0.6620561895175505),
('-2.00 < y <= -1.82', 0.004912231822041729)]
```

This shows that the coordinate with the biggest predictive power is the first one, which is the conclusion reached after observing the results obtained on the projection method.

It is easy to see a pattern in the interpretation that can be applied to every case:

- If the value in the difference vector, $\mathbf{d}$, is positive, that feature needs to increase in order for the classifier to be able to assign the point a certain class and if the value is negative, that feature needs to decrease its value.
- The bigger absolute value in $\mathbf{d}$ indicates the feature with more weight in the classification process so the changes in that feature will be more significant in the final classification than changes in other features less relevant.
- The difference between a point and its projection also indicates the closeness of the point to the decision surface. If the difference is small, the point is close to the decision boundary, if the difference is big, the point is far from the decision boundary.

This is trivial since the model is linear and its interpretation is easy. Using the proposed method, the difference between the original point $\mathbf{x}$ and its projection $\mathbf{x}'$ is a vector which is always proportional to the weight vector $\mathbf{w}$, as shown in section 3.2.

The proposed method gives as the difference between the original point $\mathbf{x}$ and its projection $\mathbf{x}'$ its always proportional to the difference vector $\mathbf{d}$, so it is trivial that the bigger the difference for a certain feature is, that characteristic is going to have more importance.

4.1.2 Non-linear model

For the non-linear model experiment, the data is generated randomly by using the method `make_moons` [4] provided by `scikit-learn`. For this example it was generated a set of 1000 points that were split into the train and test sets with the method `train_test_split` in a 90% to 10% proportion. As in the previous experiment, the model will be trained with the train set and the points explained will belong to the test set. The data obtained can be see in figure 4.3 where it can be seen two different classes.

Once the model is trained, the next step is to obtain the projection on the decision boundary using the method proposed in section 3.3. This projection is obtained for two different points and can be seen in figure 4.4. The original points are represented with a blue $\times$, the projections are represented with the red $\times$, and the $\mathbf{d}$ vector is represented by the dotted line. The colored background represent the probabilities from 0 to 1 and the decision boundary is the white band that covers the probabilities 0.45 to 0.60. The points projected are on the decision boundary so the method for projecting the points

[4] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html#sklearn.datasets.make_moons

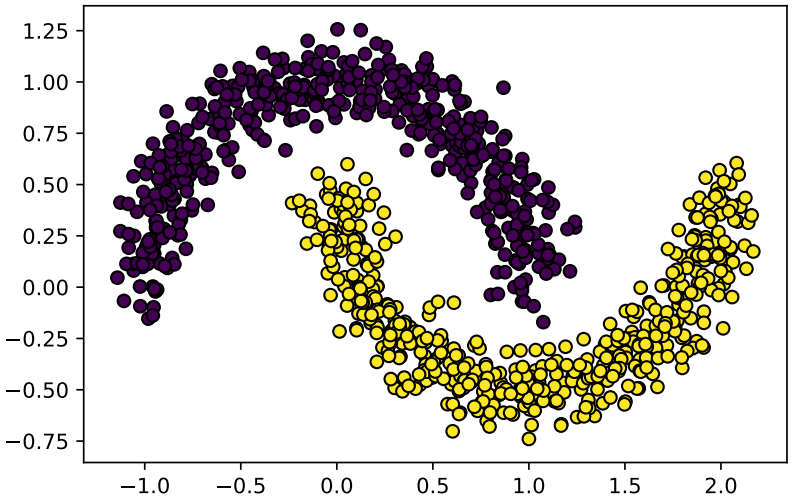


Figure 4.3: Random dataset generated for the two-dimensional non-linear experiment where the points of class 1 correspond to the yellow color and the points of class 0 correspond to the dark blue color.

behaves as it was intended.

The projection of the first point is:

```
The original point belongs to class 0.
The original point has coordinates (-0.10, 0.93).
The projection has coordinates (-0.01, 0.62).
The difference is (-0.10, 0.31).
```

The coordinates of the point are $(-0.10, 0.93)$, the coordinates of the projection are $(-0.01, 0.62)$ and the difference is $(-0.10, 0.31)$, this indicates that in order for the classifier to be able to classify the point, the first coordinate needs to be smaller and the second coordinate needs to be bigger. In absolute values, the second feature has more predictive power so the most relevant changes needs to be done in that coordinate.

The LIME interpretation of the first point is:

```
[('y > 0.68', -0.6125822832247605),
('x <= -0.09', -0.18444547516939494)]
```

Where it can be seen that the second coordinate has bigger predictive power and the first one has smaller predictive potential. This means that the coordinate that is more relevant when making predictions is the first one, which is consistent with the results obtained with the projection method.

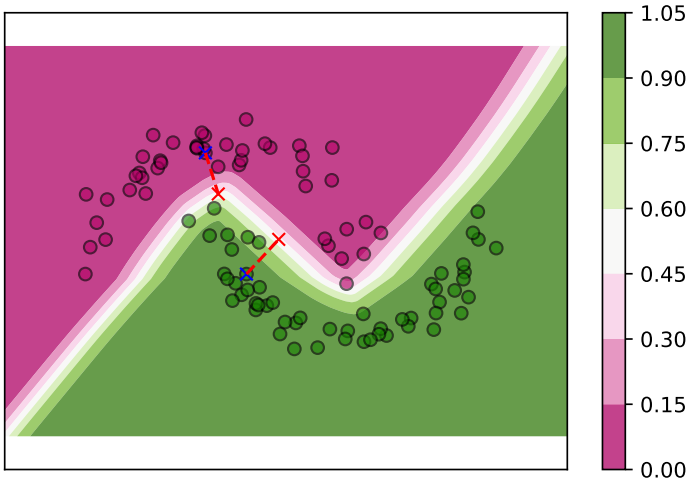


Figure 4.4: Projection of two points in the non-linear two-dimensional model. The background colors represent the probability of a point belonging to the 1 class, from 100%(green) to 0%(pink). Since there are only two classes, the points with 0% probability of belonging to the 1 class, belongs to the 0 class. The blue $\times$ represents the point that is being explained, the red $\times$ represent the projection on the decision boundary, which is the band white colored that has 50% probability of belonging to the 1 class. The dotted red line represents the difference between the original point and the projection.

Applying the projection method to the second point, the results obtained are:

```
The original point belongs to class 1.
The original point has coordinates (0.21, 0.00).
The projection has coordinates (0.46, 0.27).
The difference is (-0.25, -0.27).
```

The coordinates of the original point are $(0.21, 0.00)$, the coordinates of the projection are $(0.46, 0.27)$ and the difference between the points are $(-0.25, -0.27)$. This means that, in order for a point to be assigned the 1 class, both of the coordinates need smaller values. In absolute terms, the second coordinate is bigger so it is the most relevant coordinate in the classification process.

The LIME interpretation of the point is:

```
[('-0.20 < y <= 0.25', 0.275747838292105),
('-0.09 < x <= 0.51', 0.2443241707610095)]
```

Where it can be see that the coordinate with the bigger predictive power is the second coordinate, which is the result obtained with the projection method.

It is easy to see a pattern in the interpretation that can be applied to every case and that is equal to

the interpretation described in the linear experiment:

- For each feature, if the value in the difference vector is positive, that feature needs to increase its value in order to be classified and needs to decrease it if the value in the difference vector is negative.
- The bigger absolute value in the difference vector indicates the feature that influences the most in the classification process.
- The difference between a point and its projection also indicates how close or far the original point is to the decision surface. If the difference is small, the point is close to the decision boundary, if the difference is big, the point is far from the decision boundary.

On the non-linear case it can be seen that the difference vectors are not proportional between them, so the explanations may vary between different points, as observed on the experiments.

4.2 Low dimensionality data

For the low dimensionality test, it is used the *Iris Setosa* dataset [14]. This dataset contains 3 classes of 50 instances each one where each class corresponds to a different type of the iris plant. This is a well-studied dataset so it is known that the *Iris setosa* class is totally separable from the other two classes, so for this experiment the two classes used are going to be the *Iris setosa* and the *Iris versicolour*. Each instance of the data contains 4 features that represent the sepal length, sepal width, petal length and petal width, all expressed in centimeters, of the iris flower.

The experiments for the linear model and the non-linear model follows the same steps:

1.– Download the data.

2.– Split the data into the train and test sets.

3.– Train the model.

4.– Obtain the projection on the decision surface.

5.– Explain two points of the model.

6.– Compare the results obtained using the projection method with those obtained using LIME.

Since the first two steps are identically for both models, the data is downloaded and then split into the sets before the experiment begins. The dataset is obtained using the `load_iris` [5] provided by the library `scikit-learn`. Once the data is loaded, it needs preprocessing in order to discard the class *Iris virginica* for working only with two separable classes. When this class is discarded, the data

[5] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_iris.html

is split into the train and test datasets with the `train_test_split` method in a proportion 90% to 10% since the training set will be used for training the models and the points explained will be points belonging to the test set, as in previous experiments. When the sets are split, the experiment begins.

In other executions of the method the results may vary because, although the data are always the same, the train and the test set are generated randomly and the instances that belongs to each one may be different.

4.2.1 Linear model

The linear model experiments are made training an LR model and the method of the exact projection. When the model is trained, the projection of two points, one of each class, is obtained.

The projection for the first point is:

```
Values for the original point: (5.20,3.50,1.50,0.20) of
class setosa.
Values for the approximated projection: (5.43,3.02,2.72,0.71).
Difference: (-0.23,0.48,-1.22,-0.51).
List of features names: sepal length (cm), sepal width (cm),
petal length (cm), petal width (cm).
```

The original instance is $(5.20, 3.50, 1.50, 0.20)$, the values for the projection are $(5.43, 3.02, 2.72, 0.71)$ and the difference between them is $(-0.23, 0.48, -1.22, -0.51)$. These results indicate that the most important feature is the petal length because it is the feature with bigger absolute value. It also indicates that, in order for the classifier to know that the instance belongs to the *iris setosa* class, all the features need to decrease except the sepal width that should be bigger.

The results obtained with LIME are:

```
('1.40 < petal length (cm) <= 1.80', -0.6046921649701249),
('sepal width (cm) > 3.40', -0.046598841442598724),
('petal width (cm) <= 0.20', -0.022270409278308628),
('5.00 < sepal length (cm) <= 5.40', -0.007943838309643415),
```

and shows that the feature with the biggest predictive power is the petal length, which reinforces the interpretation of the projection since the most important feature is the one that needs bigger changes in order for the classifier to be able to classify correctly the point, as expected since the difference in the case of linear models is proportional to the weights, as explained in section 3.2.

The projection of the second point is:

```
Values for the original point: (6.10,2.80,4.70,1.20) of
class versicolor.
Values for the approximated projection: (5.77,3.49,2.94,0.47).
Difference: (0.33,-0.69,1.76,0.73).
List of features names: sepal length (cm), sepal width (cm),
petal length (cm), petal width (cm).
```

These results shows that the values of the second point are $(6.10, 2.80, 4.70, 1.20)$, the values of the projection are $(5.77, 3.49, 2.94, 0.47)$ and the difference between them are $(0.33, -0.69, 1.76, 0.73)$. This indicates that, in absolute values, the bigger feature is the petal length, so that is the feature with the biggest impact in the predictions. In order for the classifier to be able to classify the point as *iris versicolor*, all the features should be bigger except the sepal width that should be smaller.

The LIME interpretation is:

```
[('petal length (cm) > 4.20', 0.6807343673149688),
('0.45 < petal width (cm) <= 1.30', 0.01708271247796486),
('sepal length (cm) > 5.77', 0.0008804057896446728),
('sepal width (cm) <= 2.82', -0.000849432937787918)]
```

where it can be seen that the most relevant feature is the petal length, which is consistent with the results obtained by the projection method.

The same pattern described in section 4.1 can be applied in this method, since it is explaining a linear model:

- Each value in the difference vector corresponds to a feature of the instance. If the value corresponding to a certain feature is positive, the value on the projection should be bigger for the classifier to know to which class to assign the point. If the difference is negative, the value needs to decrease.
- In absolute terms, the biggest value of the difference vector indicates the feature with the biggest predictive potential so the changes in that feature would affect more to the classification process than changes in features less relevant.

Even though there were only two points explained, it is possible to have an intuition about the general behaviour of the classification. Looking at the differences between the two original points and its projections, it is possible to see that the most relevant feature is the petal length in both cases. This is because, as was demonstrated in section 3.2, the difference is proportional to the weight vector $\mathbf{w}$ and the petal length is the characteristic with the biggest weight in the full model.

4.2.2 Non-linear model

The non-linear model experiments are made training an MLP and using the method of the approximated projection proposed in section 3.3 as explained before. Once the model is trained, the experiment obtains the projection for two different points, one from each class.

The projection for the first point is:

```
Values for the original point: (5.20,3.50,1.50,0.20) of
class setosa.
Values for the approximated projection: (5.23,2.71,2.31,0.90).
Difference: (-0.03,0.79,-0.81,-0.70).
List of features names: sepal length (cm), sepal width (cm),
petal length (cm), petal width (cm)
```

The values of the original point are $(5.20, 3.50, 1.50, 0.20)$, the values of the projection are $(5.23, 2.71, 2.31, 0.90)$ and the difference between them are $(-0.03, 0.79, -0.81, -0.70)$. This indicates that all the features needs to decrease its values, except the sepal width that needs to have a larger value in order for the classifier to know that the class of the point is *iris setosa*. In absolute terms, the bigger value in the difference vector corresponds to the petal length, so that is the most relevant feature when making predictions. In this case, the difference between the original sepal length and its projection is almost zero, so that means that this feature does not need any changes in its value.

The LIME interpretation is:

```
[('1.40 < petal length (cm) <= 1.80', -0.5366120740299982),
('sepal width (cm) > 3.40', -0.1295185388973795),
('petal width (cm) <= 0.20', -0.09169969022245852),
('5.00 < sepal length (cm) <= 5.40', 0.00729981909078999)]
```

It can be seen that the most relevant characteristic is the petal length, which is the result obtained with the projection method and its the feature that influences the most in the classification process.

The projection of the second point is:

```
Values for the original point: (6.10,2.80,4.70,1.20) of
class versicolor.
Values for the approximated projection: (6.06,3.74,3.74,0.36).
Difference: (0.04,-0.94,0.96,0.84).
List of features names: sepal length (cm), sepal width (cm),
petal length (cm), petal width (cm).
```

The values for the original point are $(6.10, 2.80, 4.70, 1.20)$, the values of the projection are $(6.06, 3.74, 3.74, 0.36)$ and the difference are $(0.04, -0.94, 0.96, 0.84)$. This indicates that in order for the classifier to know to classify the point as *iris versicolor*, all the features needs to be bigger except the sepal width that needs a smaller value. In absolute terms, the feature with the biggest difference is the petal length, which is the most relevant characteristic.

The LIME interpretation is:

```
[('petal length (cm) > 4.20', 0.7132015018882838),
('sepal width (cm) <= 2.82', 0.060662424626695986),
('0.45 < petal width (cm) <= 1.30', 0.03717199527956154),
('sepal length (cm) > 5.77', 0.014032913907619881)]
```

It can be seen that the most relevant feature is the petal length, which is consistent with the result obtained with the projection method.

The same interpretation pattern as in the previous experiments can be seen:

- Each value in the difference vector corresponds to a feature. If the value corresponding to a feature is positive, the value of the projection should be bigger for the classifier to know to which class assign the point. If the value is negative, the value needs to decrease.
- The biggest absolute value of the difference vector indicates which feature has the biggest predictive potential so the changes in that feature affect the most to the classification process.

Since the problem is linearly separable, the multi layer perceptron would probably classify the points using an almost linear decision surface so the interpretation method will be similar to the linear case. The biggest difference between the two cases is that in this non-linear experiment, the difference is not proportional in both points. This means that the differences are not vectors in the same direction but they are vectors on similar directions, unlike in the linear example that the vectors were on the same directions since they where proportionals to the weight vector $\mathbf{w}$.

Even though only one point per class was explained, it appears an intuition on which are the relevant features for the classifier to make its decisions, the petal length and the sepal width, since they are the two more significant features in the explanations. That hypothesis needs to be tested with more examples in order to accept it, as the most relevant characteristic may vary between instances (something that would not happen for a linear model), but gives a general idea of the behaviour of the classifier. On the contrary, the sepal length seems to be irrelevant to the classification process since the differences between the original value and the projected one are almost non-existent.

4.3 High dimensionality data

The first experiment was a two-dimensional basic but illustrative case, that could be plotted into a two-dimensional graphic that helps the interpretation step. The second experiment was made using a four-dimensional dataset where the main features could be identified in order to facilitate the interpretation step, but this process of identifying the main characteristics may not be possible when dealing with high dimensionality datasets. However, there are experiments where the input can be visualized even if their dimensionality is high, as in this case where the input samples are images. In this section, the dataset is the dataset for optical recognition of handwritten digits provided by `sklearn`, which is a copy of the test set of the handwritten digits dataset [15] from the database UCI Machine Learning repository [16]. This dataset contains 5620 instances, each one with 64 features that could be rearranged into an 8×8 image of a handwritten digit. The instances are separated into 10 different classes, one per each digit from 0 to 9. Since the proposed projection method is aimed for binary classifiers, in these experiments the only classes used are classes 0 and 1.

The experiments for the linear model and the non-linear model follow the same steps as the previous experiments:

1.– Download the data.

2.– Split the data into the train and test sets.

3.– Train the model.

4.– Obtain the projection on the decision surface.

5.– Explain two different points of the model.

6.– Compare the results obtained using the projection method with those obtained using LIME.

The first two steps are the same for both experiments. First, the data is obtained using the `load_digit` [6] method provided by `scikit-learn`. Once it is loaded, the data needs preprocessing in order to discard the classes that will not be used in the experiments. After the preprocessing is done, the data is split into the train and test sets with the `train_test_split` method in a proportion of 90% of the instances belonging to the train set and the other 10% belonging to the test set.

4.3.1 Linear model

As in the previous experiments, the linear model experiments are made training an LR model and the exact projection method explained in section 3.2. This experiment is repeated two times in order to explain a point of each one of the classes.

[6] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html

The first point explained belongs to the 0 class as can be seen in figure 4.5 where the space is divided into a 8×8 grid that shows the image corresponding to the digit, in this case, a 0.

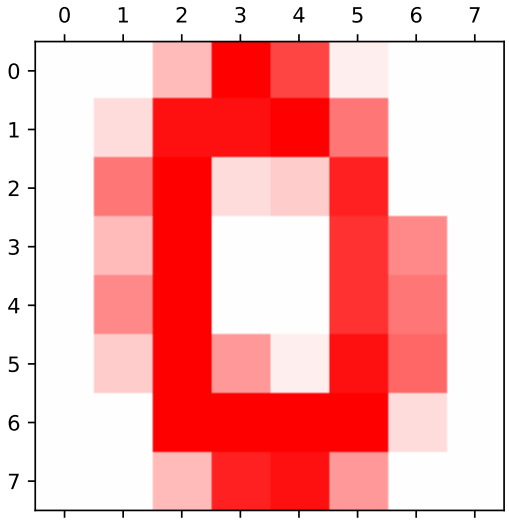


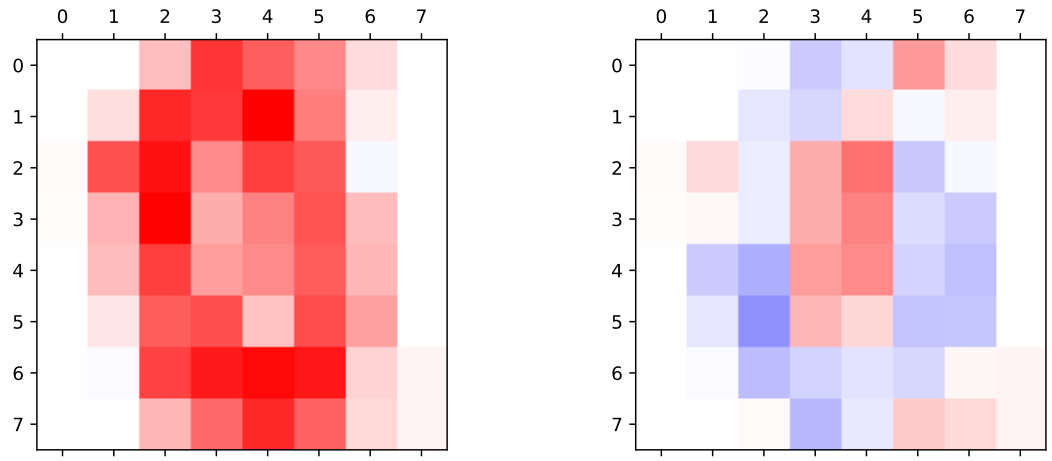
Figure 4.5: Digit belonging to the 0 class that is being interpreted.

The projection obtained for this point and the difference between the projection and the original image can be seen in figure 4.6. The projection figure 4.6(a) of the point does not provide information about the classification process but it shows a mixture between 0 and 1 which is indistinguishable even for humans. Moreover, when looking into the difference, shown in figure 4.6(b), it is easier to see that the main change between the original digit and the projection is the central region, in reddish colors, that represents the inner circle of the digit 0. This means that the inner circle of the 0 is the region of the image that makes the point be classified as a 0.

Regarding the explanation provided by LIME, figure 4.7 shows the positive and negative regions for the 0 class. On the left side of the image can be seen the regions that make the image be classified as a 0 known as the positive images, while on the left side of the image can be seen the positive regions and the regions that does not influence the classification process, known as negative regions, and that forms the original digit.

The second point explained can be seen in figure 4.8 and the projection and the difference between the original point and its projection can be seen in figure 4.9. The projection figure 4.9(a) is similar to the original digit but has some regions zero-shaped. The difference shown in figure 4.9(b) between the original digit and the projection, indicates that the main change is the central region colored in light-blue so that region needs to increase its value and the light-red region needs to decrease its value.

The projection obtained with LIME is shown in figure 4.10. In the left side of the image can be seen the positive regions that make the classifier assign the class 1 to the point and that, comparing them



(a) Projection of the first digit on the decision boundary.

(b) Difference between the original digit and the projection.

Figure 4.6: Projection and difference between the original digit 0 and the projection.

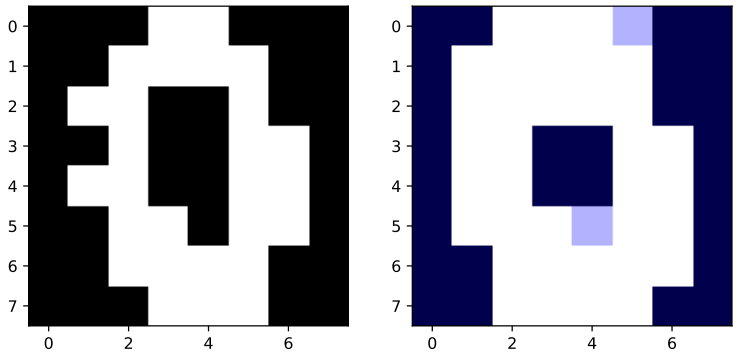


Figure 4.7: LIME explanation for the first digit. On the left side of the image it can be seen the regions that make the digit be classified as a 0, known as positive regions. On the right side of the image it can be seen the positive regions and the regions that does not contribute to the classification process, called negative regions. This right image has the same shape as the original digit.

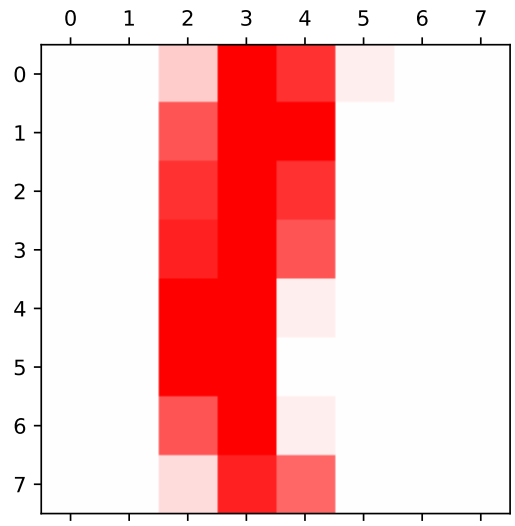
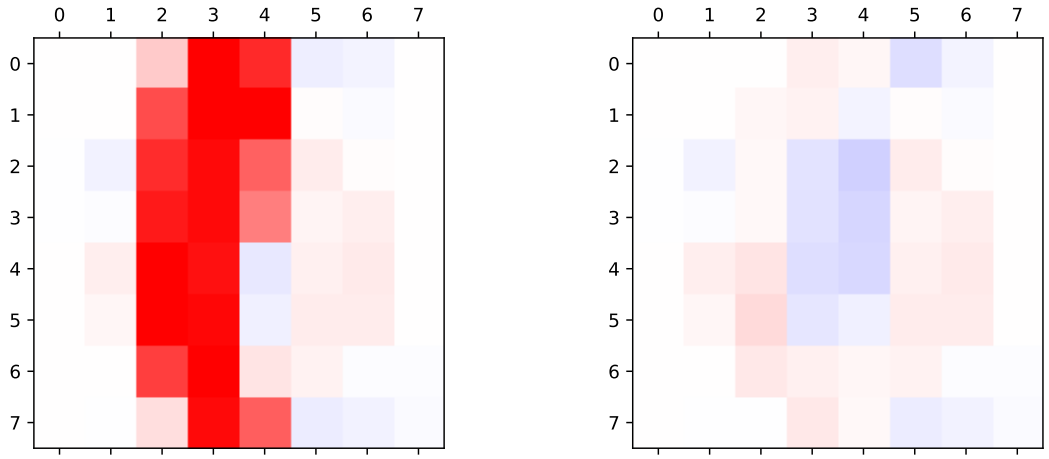


Figure 4.8: Digit belonging to the 1 class that is being interpreted.



(a) Projection of the second digit on the decision surface.

(b) Difference between the original second digit and the projection.

Figure 4.9: Projection and difference between the original digit 1 and the projection of the second point.

with the original point shown in figure 4.8, correspond to the regions with a brighter red. On the right side of the image can be seen the positive regions and the regions of the image that does not contribute to the classification process. This right image looks similar to the original digit.

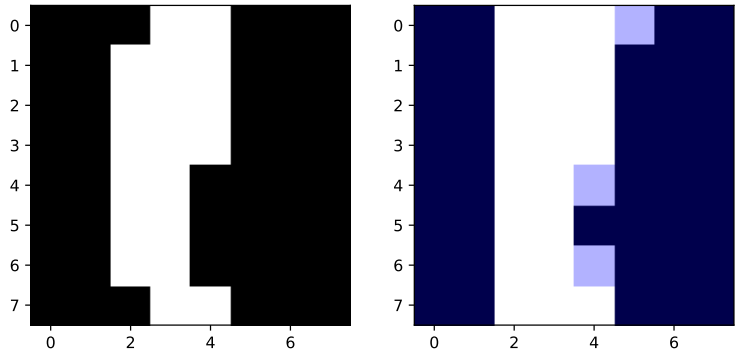


Figure 4.10: LIME explanation for the second point. On the left side of the image are shown the regions of the digit that make it be classified as a 1, known as positive regions. On the right side of the image can be seen the positive regions and those regions that does not contribute to the classification process, known as negative regions. This right image looks similar to the original digit.

In this experiment the differences between the two original digits are the same but with contrary sign. This is the result expected since for the linear model it was demonstrated in section 3.2 and in previous experiments that the difference in the linear case is proportional to the weight vector.

4.3.2 Non-linear model

As in previous experiments, the classifier chosen for the non-linear experiments is an MLP model and the method of the approximated projection proposed in section 3.3. In order to gain better comprehension of the method, two points are going to be explained, one for each class.

The first point belongs to the 0 class as it is shown in figure 4.11. Its projection, shown in figure 4.12(a), has a similar shape as the original digit. This can be also seen in figure 4.12(b) because there is no defined shape and the difference between the projection and the original digit is almost zero although it can be see a small difference, light-red colored, on the central region corresponding to the inner circle of the 0.

The LIME interpretation of the original digit can be seen in figure 4.13. On the left side of the image can be seen the positive regions of the digit, this means, the regions that contribute to the classification process. On the right side of the image it is shown the positive regions and those regions that does not contribute to the classification, called negative regions. This image looks similar to the original digit.

Since the original digit and the projection are almost the same, the positive regions of the LIME

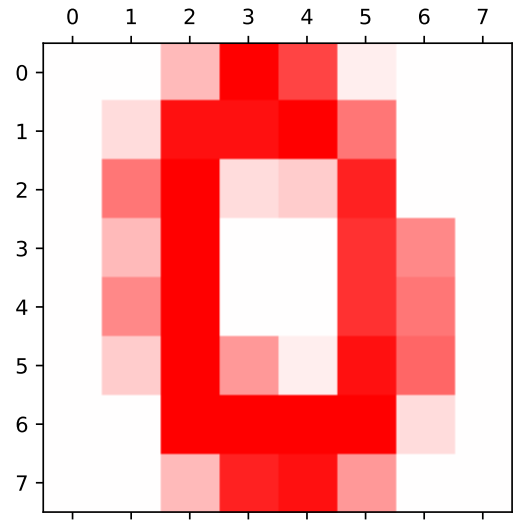
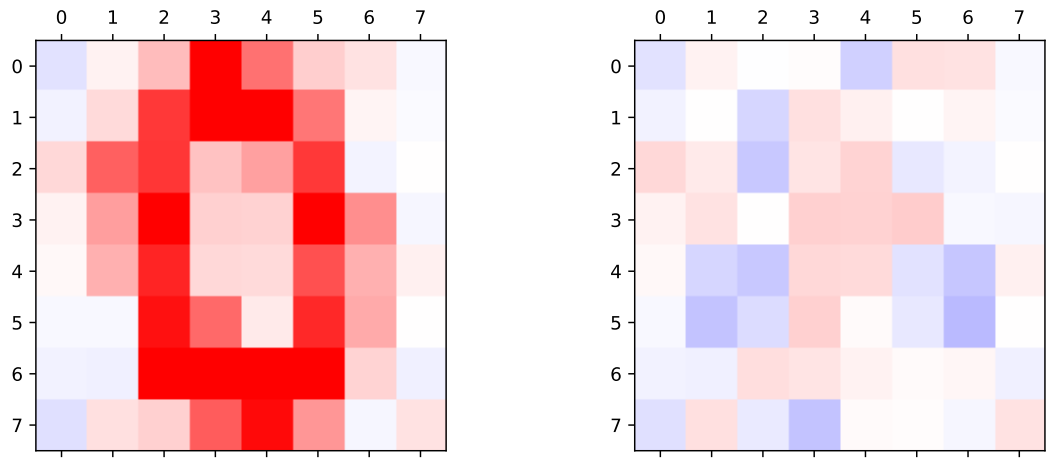


Figure 4.11: Digit belonging to the 0 class that is being explained.



(a) Projection on the decision boundary of the original digit.

(b) Difference between the original digit and its projection.

Figure 4.12: Projection and difference between the original digit 0 and the projection.

explanations matches the positive and negative regions.

This closeness between the original digit and its projection could mean that the original point that is being explained is relatively close to the decision boundary.

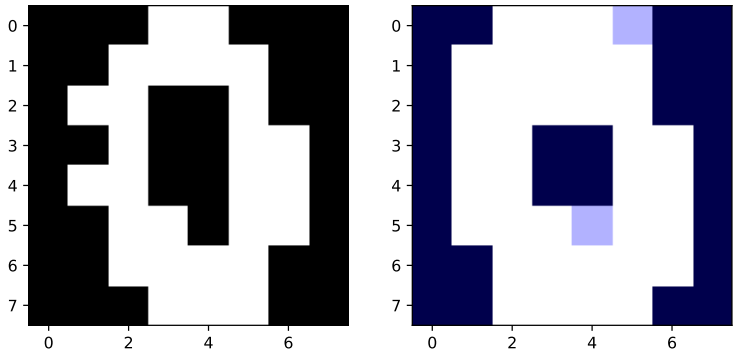


Figure 4.13: LIME explanation for the first digit. On the left side of the image it can be seen the regions that contributes to the classification process of the digit. On the right side of the image it can be seen the positive regions of the digit and those regions that does not contribute to the classification process and that are called negative regions.

The second point explained belongs to the class 1 and can be seen in figure 4.14, while its projection and the difference between them can be seen in figure 4.15. As in the previous case, the projection, shown in figure 4.15(a), has a similar shape than the original digit. This can be confirmed by looking at the difference between them shown in figure 4.15(b) because the changes between the original digit and its projection are almost zero. Nevertheless, it can be seen in light-blue and in light red, some regions that needs small changes on its values to be exactly the original digit.

The LIME explanation can be seen on figure 4.16. On the left part of the image can be seen the regions of the digit that contribute to the classification process and that are called positive. On the right side it can be seen the positive regions and the regions that does not contribute to the decision making process. This right image looks similar to the original point.

The almost zero difference of this point can also mean that the original point is relatively close to the decision boundary.

In this non-linear experiment the differences between the points and the projections are minimal. This may be because the non-linear model adjust better the data than the linear model so the original data are closer to the decision surface or even on the decision surface. In this non-linear case the differences between the two digits and its projections are not even similar because, as was explained in previous experiments and in section 3.3, both differences are not vectors on the same directions. Nevertheless, these are just preliminary experiments that should be refined to check these hypothesis.

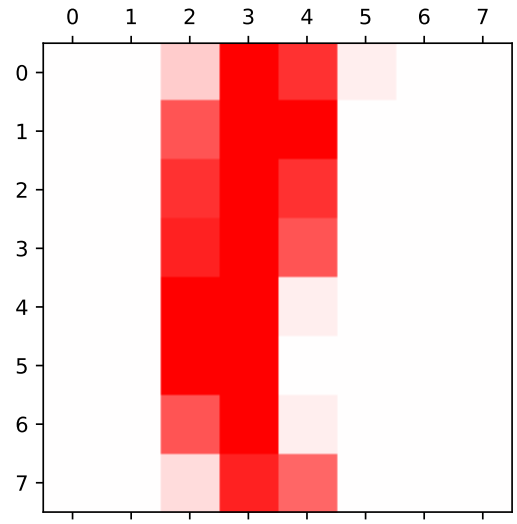
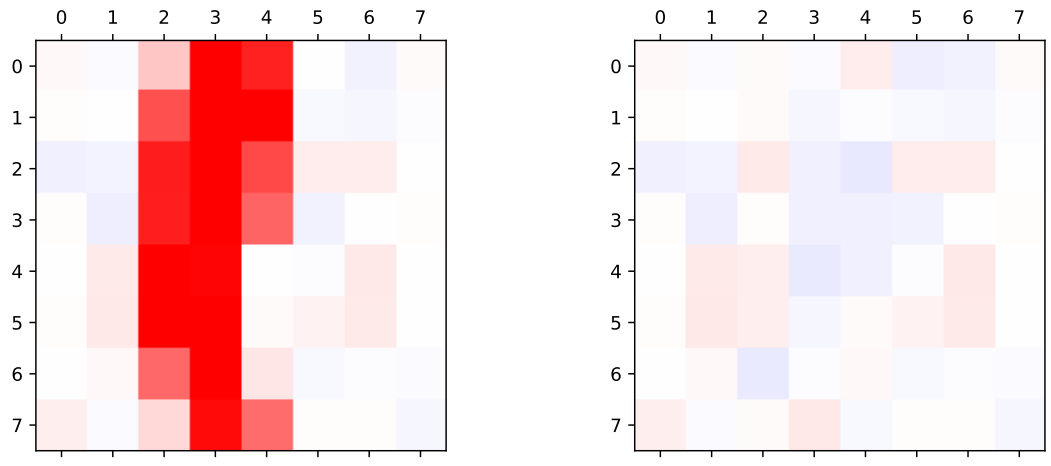


Figure 4.14: Digit belonging to the 1 class and being explained.



(a) Projection of the second digit on the decision boundary.

(b) Difference between the original digit and its.

Figure 4.15: Projection and difference between the original digit 1 and its projection.

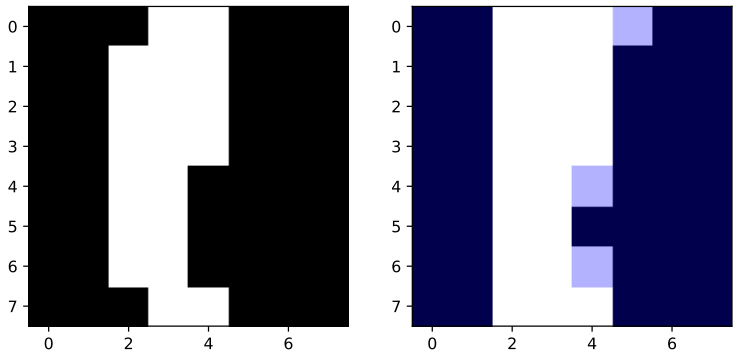


Figure 4.16: LIME explanation for the second point. On the left side of the image it can be seen the regions which contributes to classify the digit as 1, the positive regions. On the right side of the image it can be seen the positive regions and the regions, called negative regions, that does not contribute to the classification process. It looks similar to the original digit.

5

Conclusions and future work

5.1 Conclusions

To summarize, in this master thesis was done a review of IML starting with various definitions of what IML is, its properties and a taxonomy. Once the basic concepts were introduced, it was explained in more detail what are the potentially interpretable methods and three examples were given: the linear models, the decision rules and the decision trees.

Then it was done a detailed analysis of LIME, a state-of-the-art tool that can interpret any classifier since it is model-agnostic. This framework is relevant in this master thesis since it was the inspiration for the method proposed.

After the review of the IML, it was introduced the proposed method. An agnostic-model method that is able to interpret in a locally faithful way a binary classifier based on the projection of individual points on the decision boundary and analysing the differences between the point being explained and the projected point. This method has two different approaches, it can interpret a linear model using the exact equations described in section 3.2 or it can interpret a non-linear model using the approximated method proposed in section 3.3.

Once the methods are defined, they were tested over three different datasets. First it was made a base case with a two dimensional dataset generated randomly which helps to understand the concepts of the boundary, the projection and the difference between them since the data can be plotted into a two-dimensional space. The next dataset was a low dimensional dataset, the *Iris setosa*, that test the method with an example more likely to appear in real-world tasks. Finally, the last experiment was made using a dataset for optical recognition of handwritten digits, which has a high dimensionality and it is the most likely kind of problem that the method may encounter in real-life. All the experiments were made for the two methods, linear and non-linear, and then compared with the results obtained using LIME.

The results obtained are promising and the projection method was able to obtain the projection on the decision boundary of each point, provide the differences between the two points and then, based

on the differences of the point and the projection, make easier to know *why* the classifier took a certain decision and *which* are the features that help it make that decision.

5.2 Future work

Even though the results were promising, there are some points that did not reach the scope of this master thesis but would be interesting to explore since would complement the work already done.

The first and most important pending work is to propose metrics that help compare the proposal against other methods such as LIME. This would be useful since in order to know how good an interpretation method is, it is important to be able to compare it against other similar approaches.

An experiment that was not possible to make, is to try the method with more complex classifiers, as the ones based on pre-trained neural networks. This would help to test the full potential of the projection method and to understand better its behaviour against real-world problems.

It would be interesting to expand the experiments to other models in order to make a comparison table with all the models and their performances using this proposed projection method. Moreover, only three types of problems were tackled in this work, so another important future line is to test the method in different kinds of real-world problems to extend the comparison table to, not only models, but also problems. This would make possible to know with which models and problems the projection method performs the best.

Finally, another future line of work is to introduce a sparsity component that introduces sparsity in the difference between the original point and its pseudo-projection in the approximated method in order to obtain a sparse interpretation even though the number of dimensions is high, as it usually the case.

Bibliography

[1] A. Smiti, "When machine learning meets medical world: Current status and future challenges," *Computer Science Review*, vol. 37, p. 100280, 2020.

[2] I. Portugal, P. Alencar, and D. Cowan, "The use of machine learning algorithms in recommender systems: A systematic review," *Expert Systems with Applications*, vol. 97, pp. 205 – 227, 2018.

[3] Z. C. Lipton, "The mythos of model interpretability," *Queue*, vol. 16, no. 3, pp. 31–57, 2018.

[4] F. Doshi-Velez and B. Kim, "Towards A Rigorous Science of Interpretable Machine Learning," 2017.

[5] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining Explanations: An Overview of Interpretability of Machine Learning," 2018.

[6] C. Molnar, *Interpretable Machine Learning*. 2019.

[7] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should {I} Trust You?": Explaining the Predictions of Any Classifier," *CoRR*, vol. abs/1602.0, 2016.

[8] C. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[9] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*. Springer Science & Business Media, 2009.

[10] K. Patel, J. Fogarty, J. A. Landay, and B. Harrison, "Investigating statistical machine learning as a tool for software development," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, (New York, NY, USA), p. 667–676, Association for Computing Machinery, 2008.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[12] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[13] M. J. D. Powell, "Direct search algorithms for optimization calculations," *Acta Numerica*, vol. 7, p. 287–336, 1998.

[14] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[15] L. Xu, A. Krzyzak, and C. Y. Suen, “Methods of combining multiple classifiers and their applications to handwriting recognition,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.

[16] D. Dua and C. Graff, “UCI machine learning repository,” 2017.

Acronyms

IML Interpretable Machine Learning.

LASSO Least Absolute Shrinkage and Selection Operator.

LIME Local Interpretable Model-agnostic Explanations.

LM Linear Models.

LR Logistic Regression.

ML Machine Learning.

MLP Multilayer Perceptron.